# 모션 인식을 통한 쉬운 드론 제어 시스템

황진하○ 양정일 김태석 홍충선
경희대학교 컴퓨터공학과
fkrlsp2@naver.com, ya63kr@nate.com, vmfosel156@naver.com, cshong@khu.ac.kr

# Easy Drone Control System using Motion Recognition

JinHa Hwang○, Jeong Il Yang, TaeSeok Kim, ChoongSeon Hong
Dept of Computer Engineering, Kyung-Hee University

**요 약**

드론은 다양한 산업, 국방, 개인 취미 등 다양한 분야에서 적용 가능성이 높아 사회적으로 많은 관심을 받고 있다. 그러나 이러한 관심에도 불구하고 조종대 이용한 드론 조작법을 습득하기 어려워 초보자들은 쉽게 활용하기가 어렵다. 본 논문에서는 립모션 핸드 모션 인식 기기를 활용해 직관적으로 제어 가능한 드론 컨트롤 시스템을 제안한다. 제안하는 립모션을 드론 제어 시스템은 손을 사용한 쉬운 조작뿐만 아니라 사용자와 시점을 일치시켜 직관적으로 드론을 제어할 수 있다.

## 1. 서 론

현재 드론은 국방, 보안, 산업 등 다양한 분야에서 활용하기 위한 연구가 활발히 진행되고 있다. 예를 들어, 물류 분야에서 물류 운송, 농업 분야에서 파종 및 비료 작업, 재난 구조를 위한 공중 촬영을 할 수 있다. 또한 개인의 취미로 레저용, 스포츠용으로 활용하기도 한다[1].

그러나, 위와 같은 드론의 다양한 활용 처 와는 달리 개인이 드론을 소유하고 있는 경우는 많지 않다. 드론이 대중적으로 확산되지 못하는 이유 중 하나는 드론 조종의 불편함이다. 사용자가 드론을 처음 접하면 드론의 시점을 고려하여 드론의 움직임을 제어하는 어려움이 발생한다. 드론을 제어하는 연산인 피치, 요우, 스로틀, 롤을 통해 이동 방향과 시점을 조작하는 방법이 구분 되어있다. 그로 인해 드론 조종 시 사용자가 보는 시선과 일치하지 않는 괴리감으로 인해 조종에 불편함을 느끼게 된다. 드론 조종에 숙련된 경우에도 사람이 계속 조종대 통해 이동할 위치를 정해야 하기 때문에, 컨트롤러에 주기적인 물리적 압력을 가해야 한다. 따라서 이러한 결과로 오랜 시간 드론을 조종하기 어렵다.

본 논문에서는 립모션(Leap Motion) 핸드 모션 인식 기기를 활용해 직관적으로 제어 가능한 드론 컨트롤 시스템을 제안한다. 제안하는 립모션을 드론 제어 시스템은 손을 사용한 쉬운 조작뿐만 아니라 사용자와 시점을 일치시켜 직관적으로 드론을 제어할 수 있다.

## 2. 관련 연구

### 2.1 드론

드론은 기체에 사람이 타지 않고 지상에서 무선 전파 유도에 의해서 원격 조정이 사전에 입력된 프로그램에 따라 비행이나 조정이 가능한 무인 항공기를 말한다. 초기의 목적은 표적, 정찰, 감시 등의 군사용으로 개발되었다. 이후 점차 활용 목적이 민간 분야로 확대되어가고 있으며 화산 지역, 자연재해 지역, 원자력 발전소 사고지역 등의 인간이 접근하기 어려운 지역으로 드론을 투입하여 운용하기도 한다.

드론을 제어하는데 있어서 기본적으로 두 가지 방법이 있다. 일반적인 방법으로 많이 활용되는 조이스틱을 통한 조종과 최근에 많이 쓰이는 스마트폰 앱을 통한 조종이 있다. 대부분의 드론 컨트롤러는 드론의 시점과 고도를 제어하는 스틱과, 드론의 시점을 기준으로 전후 좌우를 이동해 컨트롤한다.

드론을 제어하는 주요 연산은 표 1과 같이 복잡하여 사용자의 제어를 어렵게 한다.

표 1. 드론 조종 연산

| 연산 명 | 기능 |
|---|---|
| 상하타(Throttle, 스로틀) | 드론을 고도를 조종 |
| 전후타(Pitch, 피치) | 드론의 전진과 후진 조종 |
| 회전타(Yaw, 요우) | 제자리에서 반시계, 혹은 시계방향으로 회전 조종 |
| 좌우타(Roll, 롤) | 드론이 좌측, 혹은 우측으로 기울이며, 그 방향을 전진 |

### 2.2 립모션 모션 인식

립 모션은 샌프란시스코에 있는 벤처회사이자, 매우 정교한 손 동작 인식 센서를 지닌 유저 인터페이스를 제공하는 장치[2]를 말한다. 작은 아이팟 크기의 장치로 8입방 피트(cubic feet)의 3차원 공간을 매우 정확하게 읽어낸다.

## 3. 립모션을 이용한 드론 제어 시스템 설계

### 3.1 시스템 구조

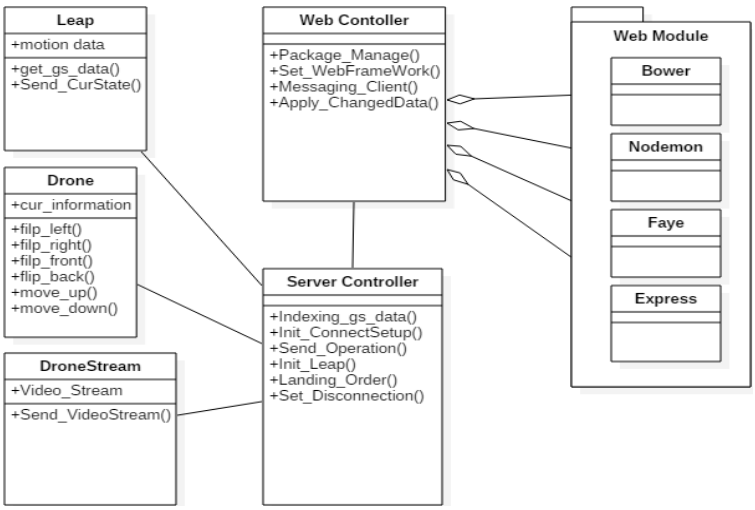


그림 1 시스템 구조

제안하는 시스템은 모션 데이터를 처리하여 드론 명령어로 가공, 드론과 무선 웹 통신을 통해 연결하여, 해당 처리 결과를 사용자에게 보여준다. 시스템은 Leap, Drone, DroneStream, Web Controller, Server Controller, Web Module로 구성한다.

Server Controller는 Leap 모듈에서 모션 데이터를 받아, 이를 처리하여 Drone에 명령어를 전송한다. Web Controller는 Web Module을 사용해, 웹 환경을 구상한다. Leap Module은 현재 입력 받은 모션 데이터[3]를 Server Controller에게 전송한다. Drone Module은 Server Controller의 명령어를 받아, Drone 제어 명령어[4]를 제공한다. DroneStream은 드론이 촬영한 영상에 대한 정보를 저장한다.

제안하는 시스템의 드론 제어 과정은 다음과 같다. Default State인 Drone을 제어하기 위해, Server와 연결을 한다. 연결이 성공적으로 완료 되면 Drone은 Connected State가 된다. Connected State인 Drone을 이륙하기 위해 Floating Motion을 LeapMotion을 통해 입력한다. 입력 된 Motion Data는 Server Application을 통해 Drone에 float 명령을 전송한다. Floating State인 Drone의 움직임을 제어하기 위해, LeapMotion에 Drone control motion을 입력한다. Drone은 Server로부터 받은 명령에 따라 정해진 비행을 하거나, 사진을 촬영하는 등 주어진 기능을 수행한다. Floating State인 Drone을 Landing 시키기 위해, 해당 Motion을 입력한다. 해당 Server는 드론의 안전한 착륙을 보조하며, Drone은 Landing 기능을 수행한다. 이후, 해당 기능을 마친 Drone과 Server의 연결을 해지한다.

### 3.2 드론 제어

피치, 요우, 스로틀, 롤을 통한 드론 제어로 인해 드론의 이동 방향과 시점을 조작하는 방법이 구분 되어 있어, 드론 조종 시 사용자가 보는 시선과 일치하지 않는 괴리감이 발생한다. 그림 2는 이에 대한 예시이다. 전진 피치 명령을 전송하였을 때, 드론의 조종자의 시선에 따른 전진이 아닌 드론의 시점을 기준으로 한 전진을 하게 되어 조종자의 시선과 일치하지 않게 된다.

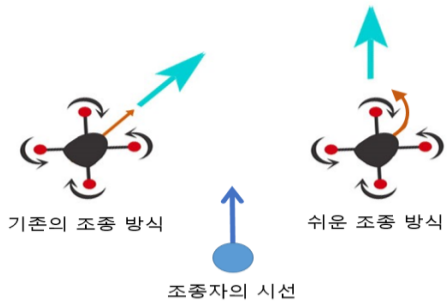


그림 2 조종 방식

이와 같은 문제를 해결하기 위해 피치와 롤을 조작 시 드론의 요우를 진행 방향과 일치시키는 작업을 다음과 같이 제안한다. Server에서 후진 피치 명령어를 전송할 때, 180° 회전 요우 명령어를 전송한다. 롤 명령어를 전송 시, 우측 이동 방향의 경우 90° 요우 명령어, 좌측 이동 방향의 경우 -90° 요우 명령어를 전송한다. 회전에 대한 속도는 각 움직임 명령어를 전송하였을 때, 사용자가 회전 하는 것을 감지할 수 있도록, 수초 내로 동작하도록 한다. 본 시스템에서 기본값은 1초로 설정한다. 이를 통해 그림 2의 쉬운 조종 방식과 같이 드론은 항상 조종자의 시선과 일치하는 방향으로 전진한다. 또한 이 때 드론의 시점 또한 자동으로, 진행방향에 맞춰 회전한다. 따라서 드론 사용에 익숙지 못한 사용자는 기존의 자동차, 혹은 비행기처럼, 조종 방향과 시점이 일치하여, 더 직관적으로 드론을 제어 할 수 있다.

앞서 설명한 직관적인 진행방향 및 시점 일치 방법을 립모션 핸드 트래킹을 이용해 설계한다. 립모션을 통해 얻은 8입방 피트의 3차원 공간의 환경에서 손의 움직임을 감지하여 얻은 모션 데이터를 통해 제어 한다. 드론에 대한 움직임 방향 조종은 3차원 공간의 XY 평면의 중심을 기준으로 구현하며, 손의 중심의 방향으로 드론의 진행방향을 결정한다.

드론의 움직임에 대한 출발/정지에 대한 조작은 손의 모양을 통해 제어하게 되며 기본 상태는 드론이 움직일 수 있는 상태를 유지한다. 이후 주먹을 쥐게 되면 드론

을 정지한다. 손의 모양에 대한 인식은 엄지 손가락을 제외한 손가락 마디 끝의 Finger Data와 손의 중심 사이의 거리를 통해, 주먹의 움직임을 인식한다. 손가락 마디 끝의 좌표가 각각 좌측 검지 손가락부터 $(x_0, y_0, z_0)$, $(x_1, y_1, z_1)$, $(x_2, y_2, z_2)$, $(x_3, y_3, z_3)$이라 하고, 손의 중심의 좌표가 $(x_m, y_m, z_m)$ 이면, 손가락 끝의 중심점의 좌표를 식 (1)과 같이 계산한다.

$$(x_{avr}, y_{avr}, z_{avr}) = \sum_{i=0}^{3}(x_i, y_i, z_i) \quad (1)$$

그리고 식 (1)을 통해 좌표 사이의 거리를 식 (2)와 같이 계산하여 해단 손의 모양을 감지한다. 본 시스템에서 손의 모양을 주먹으로 인식하는 기본 값은 5cm로 설정한다.

$$d = \sqrt{(x_{avr} - x_m)^2 + (y_{avr} - y_m)^2 + (z_{avr} - z_m)^2} \quad (2)$$

위의 조종 시스템을 적용 시, 드론의 비행 도중 조종자의 시점이 변경되면 조종사의 시선과 드론이 기존에 가지고 있던 조종사의 시선의 차이로 조종에 어려움을 겪는다. 따라서 이를 해결 하기 위해, 손의 모양을 통해 기존의 시스템이 저장하고 있던 조종자의 시점을 현재 드론의 시점으로 업데이트 한다. 해당 동작은 검지와 중지를 모두 피고, 엄지를 제외한 나머지 손가락을 전부 오므린 동작이다. 해당 동작에 대한 인식은 먼저 검지 손가락과 중지 손가락의 좌표의 중심 $(x_i, y_i, z_i) = \{(x_0, y_0, z_0) + (x_1, y_1, z_1)\}/2$를 계산한다. 이후 왼손 약지와 소지의 중심 $(x_k, y_k, z_k)$를 계산한다. 이를 통해 손의 중심 $(x_m, y_m, z_m)$ 과 검지와 중지의 중심 좌표는 일정 거리 이상 바깥에 존재하는지, 약지와 소지의 중심 좌표는 일정 거리 이상 안쪽에 존재하는지에 대한 여부를 이용해 해당 모션을 인식한다. 본 시스템에서 사용하는 거리 값은 각각 7cm, 3cm를 사용한다.

## 4. 시스템 구현

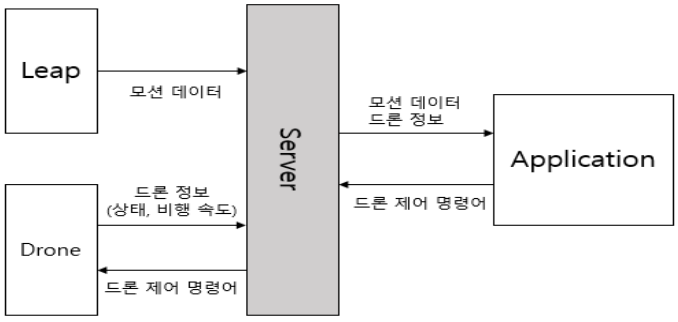


그림 3 구현물의 조종 방식

그림 3은 제안하는 시스템의 구조이다. 립모션을 통해 모션 데이터를 입력하고, 모션 데이터를 어플리케이션에 전달한다. 드론 또한 드론의 상태를 주기적으로 서버에 전달해, 어플리케이션에서 처리할 수 있도록 한다. 어플리케이션은 입력한 모션 데이터를 해독하고 현재 드론의 정보를 이용해 드론 명령어를 구성한다. 구성한 드론 명령어를 서버를 통해 드론에 전달한다. 제공한 드론 명령어로 드론이 사용자가 원하는 동작을 실행한다. 해당 시스템은 LM-0102 모델(립모션)과 Parrot BEBOP 2 FPV 모델(드론)을 사용하여 구현한다. 어플리케이션 구현은 JavaScript를 이용하였고, 서버 구현은 windows 10의 환경에서 Node.js 플랫폼을 이용해 구현한다.


그림 4 립모션을 통한 드론 제어

## 5. 결 론

본 연구를 통해, 모션 인식을 이용하여 손의 움직임으로 드론을 제어한다. 또한 사용자와 드론의 시선을 일치시키는 솔루션을 통해 기존의 드론의 시점과 사용자의 시점이 일치하지 않아 생기는 문제는 해결한다. 따라서 해당 연구를 통해, 드론 사용에 익숙지 않거나 드론을 처음 조종하는 조종자에게 더 쉬운 조종 환경을 제공한다.

* 본 연구는 미래창조과학부 및
정보통신기술진흥센터(IITP)에서 지원하는
서울어코드활성화지원사업(2011-0-00883)과
SW중심대학지원사업(2017-0-00093)의 지원으로
수행되었음.

## 6. 참고문헌

[1] 장성기, 『드론 새로운 세상을 만나다』, 크라운(2016)
[2] 우지윤, 『스크래치 아두이노 립모션』, 디지털북스(2016)
[3] Google, “Leap Motion Developer”, https://developer.leapmotion.com/, (2017.3.27)
[4] Google, “Parrot Dev”, http://developer.parrot.com/docs/SDK3/ (2017.03.27)